

(19)



Europäisches Patentamt

European Patent Office

Office européen des brevets



(11)

EP 0 898 235 A2

(12)

EUROPEAN PATENT APPLICATION

(43) Date of publication:

24.02.1999 Bulletin 1999/08

(51) Int. Cl.⁶: G06F 17/20

(21) Application number: 98114462.9

(22) Date of filing: 30.07.1998

(84) Designated Contracting States:

AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU
MC NL PT SE

Designated Extension States:

AL LT LV MK RO SI

(30) Priority: 31.07.1997 US 54366 P

01.12.1997 US 982308

(71) Applicant: AT&T Corp.

New York, NY 10013-2412 (US)

(72) Inventors:

- Douglls, Frederick
Somerset, New Jersey 08873 (US)
- Haro, Antonio
Paterson, NJ 07524 (US)
- Rabinovich, Michael
Gillette, NJ 07933 (US)

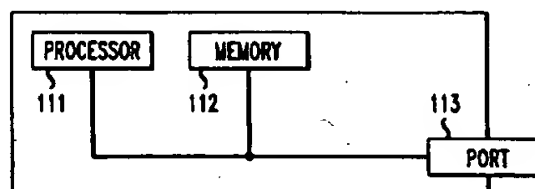
(74) Representative:

Modiano, Guldo, Dr.-Ing. et al
Modiano, Josif, Pisanty & Staub,
Baaderstrasse 3
80469 München (DE)

(54) Method and apparatus for dynamic data transfer

(57) A system and method for transferring information resources over a network from one source to another that reduces latency and bandwidth requirements on the network. The information resources are made up of at least a static and dynamic portion. A client requests information, and a server receives the request and then transmits to the requestor the desired information. The server may send both the static and dynamic portion of the information resource, or the server may send only the dynamic portion of the resource, depending on the client's needs and the request made. By discriminating between the static and dynamic portions of an information resource in this way, less data must be sent from the server to the client on every access. Also, the computational load that is typical of servers on a network is shifted to the client, thereby reducing latency.

FIG. 1B



EP 0 898 235 A2

Description**CROSS REFERENCES TO RELATED APPLICATIONS**

5 [0001] This application claims the benefit of U.S. Provisional Application No. 60/054366, filed July 31, 1997.

BACKGROUND OF THE INVENTION**FIELD OF THE INVENTION**

10

[0002] The present invention relates to information over a network, and in particular, ways of transferring resources over a network to reduce latency and bandwidth requirements for the network.

[0003] There are several standard methods of data transfer that are used to decrease latency and bandwidth requirements on a network. Three examples are caching, compression, and delta encoding. Caching is a way of temporarily
 15 storing a resource, and plays a crucial role in a wide-area distributed system such as the World Wide Web. For example, requested web pages can be quickly retrieved from a web page cache rather than the original server, saving time and network traffic. It significantly reduces response time for accessing cached resources by eliminating long-haul transmission delays. Furthermore, caching reduces backbone traffic and the load on content providers. However, caching is of limited value when much of the data on a network is dynamic, such that information is likely to change after the data
 20 was last cached. Under these circumstances, the cached information is no longer reliable. Therefore, when a significant portion of network resources are dynamic, they are not cacheable because the resource is freshly generated upon every access. In addition, the content provider might explicitly prohibit caching.

[0004] Compression is a way of reducing data size to save space or transmission time by eliminating redundancy. For data transmission, compression can be performed on just the data content or on the entire transmission (data content
 25 plus header data). However, the amount a file can be compressed is limited by the redundancy in the file content, typically by 50% for text.

[0005] Delta encoding is a way of transmitting encodings of the changes between subsequent versions of a resource. This avoids having to resend the entire resource each time it is changed. Instead, only the changes are sent. This reduces bandwidth requirements and improves end-to-end performance. In order to use delta encoding, a complete
 30 version of a resource is cached in a first computer, the one sending the request, but for the first computer to view that cached resource, the cached resource must be compared with another complete version of that resource cached or stored in a second computer, the computer responding to the request. The difference between the two complete resources is then computed, and the older cached resource is updated and then displayed. With delta-encoding, one might cache a resource even if it is considered uncacheable, but not present the cached data without first obtaining the
 35 changes to its current version. One advantage of this method is that it applies uniformly to all uncacheable resources regardless of the reason why the resources are uncacheable. Another advantage is that delta encoding can be implemented transparently, via proxies, so that content providers need not be modified. However, the delta-encoding proposals have disadvantages as well. Delta-encoding requires both computers to have a common version base used by the sending computer to compute the delta and by the receiving computer as the basis against which to apply the delta. If
 40 the content provider must compute the delta-encodings on the fly, it suffers overhead and must store a potentially large number of past versions; if the delta computation is performed by an intermediary, then the entirety of each version of the resource must be sent from the content provider to the intermediary, and the encoding must still be performed on the "critical path."

[0006] With a common class of resources, such as those provided by search engines, a significant part of the
 45 resource is essentially static. Portions of the resource vary to different extents from one query response to another (the difference between two pages in response to a single query is usually smaller than the difference between pages from different queries). Also, the location of the dynamic portions relative to the rest of the resource does not change. These resources traditionally have been dealt with in a calculation-intensive and memory wasteful environment.

[0007] For example, a technique called Server-Side Includes (SSI) allows one to specify a placeholder on a page
 50 where a dynamic content can be inserted. However, this technique is performed on the server in a calculation-intensive way. Another example is Microsoft's Active Server Pages (ASP). This technique includes instructions to pre-process a page. However, this pre-processing again is performed by the server, so bandwidth or server load are not reduced. The present invention seeks to overcome these shortcomings.

SUMMARY OF THE INVENTION

[0008] The present invention introduces a way of reducing a network's latency and bandwidth requirements in a way different from previously-known methods. The benefits of this invention are achieved by caching static portions of the

dynamic resource on the client and by shifting much of the computational burden from the server to the client.

[0009] The invention involves a regime which allows the explicit separation of static and dynamic portions of a resource. In other words, the resource is explicitly separated into a template and per-query bindings; the client can pre-process them in order to arrive at a current instantiation of the resource. The static portion may contain macro-instructions for inserting dynamic information. The static portion which includes these instructions (the *template*) can be cached freely. The dynamic portion contains the bindings of macro-variables to strings specific to the given access. The bindings are obtained for every access, and the template is expanded by the client prior to rendering the document. In this way, and unlike delta encoding, only one version of the resource is necessary in order to reconstruct a complete and updated resource. Also, the server must generate the dynamic data only for each access, not the entire resource.

[0010] The present invention is advantageously designed to minimize the size of the bindings. Experiments have shown a remarkable difference in size between the original resource and the bindings. According to our latest data, the uncompressed bindings are 4-8 times smaller than the uncompressed original resource, while compressed bindings are 2-4 times smaller than the compressed original resource. Thus, sending just the bindings results in the corresponding traffic reductions compared to sending entire resources.

[0011] Although the terms "client" and "server" have well-understood meanings in the context of the Internet, for purposes of this invention, this description uses the terms in a broader sense: the term "client" means an entity that requests information, and the term "server" means an entity which receives a request for information.

[0012] In one embodiment of the present invention, the client requests from the server the entire resource, and the server responds by sending only the dynamic portion of the document, which is reconstructed by the client. If the client does not have the static portion, a request is sent to the server, and the server responds by supplying the static portion of the resource, which can be cached for subsequent access.

[0013] In another embodiment, the client will first determine whether it needs the static portion of the document. If it does, it first requests both the static and dynamic portion of the resource. If the client determines it does not need the static portion, it simply requests the dynamic portion.

BRIEF DESCRIPTION OF THE DRAWINGS

[0014]

Figure 1a illustrates a system-level block diagram of an embodiment of the present invention.

Figure 1b illustrates an embodiment of an apparatus and system in accordance with the present invention.

Figure 2 illustrates a flow chart of an embodiment of a method in accordance with the present invention.

Figure 3 illustrates a flow chart of another embodiment of a method in accordance with the present invention.

Figure 4 illustrates a flow chart of another embodiment of a method in accordance with the present invention.

Figure 5 illustrates a flow chart of another embodiment of a method in accordance with the present invention.

DETAILED DESCRIPTION

[0015] The present invention is directed to ways of transferring resources over a network to reduce latency and bandwidth requirements for the network. Referring now in detail to the drawings, there is illustrated in Figure 1a a system-level embodiment of the present invention. Clients, in this figure denoted by 101, 102, and 103, request through network 100, at least a portion of a resource from the server 110. The resource presumably is made up of at least a static portion and a dynamic portion. The server then supplies either the entire resource, just the dynamic, or just the static portion of the resource, depending on the client's needs. Since the static portion can be cached by the clients, only the dynamic portion is usually supplied for popular resources, thereby reducing the network traffic and the load of the content server.

[0016] Typical examples of requests for resources are Uniform Resource Locators (URLs). In an embodiment of the present invention, the resource requested is a document written in Hypertext Markup Language (HTML) containing both static and dynamic portions, although the spirit and scope of the invention is not limited to HTML. The static portion, or template, contains tags which delineate where in the HTML code the components of the dynamic portion are to be inserted. Tables 1 and 2 show an example of how to delineate the static and dynamic portions of the data.

[0017] Consider the following resource:

```

<HTML>
<HEAD>
<TITLE>Michael Rabinovich</TITLE>
<BODY>
...
The time is 1:15pm.
This page has been accessed 10 times.
</BODY>
</HTML>

```

[0018] In Table 1, the left-hand column contains the template, or static portion, of this resource. The right-hand column contains the dynamic portion of the resource at issue corresponding to a particular instantiation. In Table 1, the static portion of the data contains the tag "VAR" and variables "time" and "count." The "VAR" tags delineate where in the HTML the dynamic data corresponding to the variable "time" and "count" are to be inserted. In the example in Table 1, the "VAR" tag is used to define a single dynamic variable and is used in the template as a placeholder for a variable to be replaced by a text segment dynamically bound to this variable. The tag "DYNAMICS" is used to define the dynamic portion of the resource. This dynamic portion, which always corresponds to a particular instantiation of the resource, is stored in a way that can be accessed by the server and sent to the client. The client then has the burden of combining the static and dynamic portions of the resource by inserting the instant values, in this case "time = 1:15pm" and "count = 10," into their proper places in the static portion of the resource so that the entire resource may be viewed by the client.

Template for a simple query.	The dynamic portion corresponding to particular instantiation.
<pre> <HEAD> <TITLE>Michael Rabinovich</TITLE> <BODY> ... The time is <VAR time>. This page has been accessed <VAR count> times. </BODY> </pre>	<pre> <HTML> <TEMPLATE HREF="query.hpp"> <DYNAMICS> time = 1:15pm; count = 10; </DYNAMICS> </HTML> </pre>

Table 1

[0019] In another embodiment of the invention, a tag in the template allows the template, possibly in combination with one or multiple dynamic-variable tags like "VAR," to loop through a calculation. A pre-processor would expand the template fragment within the loop as many times as the number of bindings to the macro-variable specified in the loop frag-

ment of the dynamic part of the resource. This is illustrated in Table 2. Furthermore, the loop may contain multiple dynamic variable tags. When the loop has multiple dynamic variables, each iteration consumes the next dynamic binding for each dynamic variable. When the dynamic data list is exhausted for one of the variables, the loop stops.

[0020] Consider the following resource, which is the original resource for Table 2:

```
<FONT SIZE=+1>1</FONT>  
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&~  
<A HREF="/cgi-bin/search-  
engine?first=1&query=caching+dynamic+objects">2</A>  
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&~  
<A HREF="/cgi-bin/search-  
engine?first=2&query=caching+dynamic+objects">3</A>  
<...>
```

[0021] The left-hand column of Table 2 contains an example of a template of this resource in the context of a hypothetical search engine called "search-engine." The template contains the dynamic-variable tag "LOOP" with the dynamic-variable tag "VAR" nested within the loop. After increasing the font size and outputting "1," the static portion of the resource shown in the left-hand column of Table 2 begins a loop. Every operation within the tags <LOOP> and </LOOP> is performed a variable number of times set by the dynamic portion of the resource, shown in the right-hand column of Table 2.

Table 2

[illegible]

[0022] In another embodiment of the present invention, conditional statements, e.g. "IF," are used within the loop variable. In another embodiment of the present invention, tags are introduced to set or define an initial or subsequent value of a dynamic or iterative variable.

[0023] Along these lines, an embodiment of the present invention is implemented as an extension to HTML. The HTML extension contains the following new tags: VAR, LOOP, IF, SET-VAR and DYNAMICS. The VAR and LOOP tags are used in the template as instructions regarding the location of dynamic content. In other words, they are used as macro-expressions in the HTML text to mark the place in the static portion of the resource where dynamic data may be inserted. The DYNAMICS tag is used to define the dynamic part of the document. It contains bindings of the macro-variables to dynamic text strings specific to the current document access. The IF tag is a conditional statement, and the SET-VAR tag is used to define an initial or subsequent value of a dynamic variable in the context of iterative calculations. The LOOP tag is a way of performing iterative calculations with dynamic variables. As can be seen with these tags, and unlike delta encoding, only one version of the resource is necessary to create the entire resource. A client that has the ability to use the present invention may relay that information to the server via the HTTP header. Of course, the names of the tags are arbitrary and are chosen in this case simply for their descriptive value; the tags may take any name as

long as they function in the desired way.

[0024] Figure 1b shows an embodiment of an apparatus in accordance with the present invention. The apparatus in this embodiment corresponds to server 110 in Figure 1a, and includes a processor 111, a memory 112 that stores instructions adapted to be executed in the processor 111 to receive requests for resources and to send resources, and a port 113 adapted to be connected to the network, with both the port and memory coupled to the processor. In this embodiment, a server 110, connected to a network through a port, would store in memory instructions necessary to respond to requests for resources. After a request for at least a portion of a resource enters the apparatus from the network via the port, the processor responds to the request by sending back through the network either the dynamic portion of the data or both the static and dynamic portions of the data. In one embodiment, there is a memory for storing at least a portion of the resource.

[0025] Figure 2 shows an embodiment of a method in accordance with the present invention, shown from the server's perspective. At step 201, a server receives a request for an entire resource, having at least one static portion and one dynamic portion. At step 202, in response to this request, the server sends only the dynamic portion of the resource. This dynamic portion contains data allowing the client to find and obtain the static counterpart. At step 203, if the server then receives a request for at least the static portion, the server sends the static portion in response to the request.

[0026] Figure 3 shows an embodiment of a method in accordance with the present invention from the client's perspective of the transaction. At step 301, the client requests an entire resource, having at least one static and one dynamic portion. At step 302, the client then receives, in response to this request, only the dynamic portion of the resource. This dynamic portion contains data allowing it to find and connect to its static counterpart. At step 303, if the client determines that it needs the static portion, e.g., the static portion is not cached at the client, the client then moves to step 304, requesting from the server the static portion of the resource to match up to the dynamic portion of the resource, and step 306, where the client receives the static portion. After use of the data resource, the static portion is cached to await future use. At step 305, after the client has both the static and dynamic portions, the next step is to combine the dynamic and static portions.

[0027] In another embodiment of a method in accordance with the present invention, shown in Figure 4 from the server's perspective, the party desiring the resource (the client) at step 401 determines, as an initial matter, whether it needs the static portion of the resource. This differs from the previously-described embodiment in that, in the previously-described embodiment, the client requests the entire resource and the server responds with simply the dynamic portion. However, in the embodiment shown in Figure 4, the client makes an initial determination of its needs. At step 402, if the client does not need the static portion, the server receives a request for at least the dynamic portion of the resource, and in response, at step 403, sends the dynamic portion of the resource. If the party does need the static portion, at step 404, the server receives a request for at least the static and dynamic portions and at step 405 sends both static and dynamic portions. This embodiment is advantageous because it reduces the total number of requests made to a given server, thereby reducing total transmission time.

[0028] Figure 5 shows an embodiment of the present invention from the client's perspective of the transaction. From the client's perspective, the party desiring the resource (the client) determines up front at step 501 whether it needs the static portion of the resource. If the client does not need the static portion, then at step 502, the client sends a request for at least the dynamic portion of the resource, and in response, at step 503, receives the dynamic portion of the resource. If the client does need the static portion, at step 504 the client requests at least the static and dynamic portions and at step 405 receives both static and dynamic portions. Finally, at step, 506, the client combines the static and dynamic portions.

[0029] As explained in detail above, the invention allows for data transfer with reduced bandwidth and latency requirements in a network. The major benefit is found in a server that is configured to receive requests for dynamic data, or portions thereof, and respond as detailed above. With this invention, the server's computational burden is reduced by eliminating the necessity of generating the entire resource, comparing it to an earlier version of the same resource and computing the difference. Furthermore, unlike the prior art, only one version of the resource is necessary to create a new instantiation.

[0030] Although various embodiments are specifically illustrated and described herein, it will be appreciated that modifications and variations of the present invention are covered by the above teachings and within the purview of the appended claims without departing from the spirit and intended scope of the invention. For example, while embodiments of the invention are useful with dynamic documents written in HTML, it can be appreciated that any number of other languages, such as SGML, can be used with the present invention.

[0031] Where technical features mentioned in any claim are followed by reference signs, those reference signs have been included for the sole purpose of increasing the intelligibility of the claims and accordingly, such reference signs do not have any limiting effect on the scope of each element identified by way of example by such reference signs.

Claims

1. A method for transferring over a network at least a portion of a resource having at least a static and dynamic portion, comprising the steps of:
 - a. receiving a request from a client for at least a dynamic portion of the resource;
 - b. sending the dynamic portion of the resource, the dynamic portion adapted to be combined with the static portion of said resource; and
 - c. if said client lacks the static portion, then performing the sub-steps of:
 - i. receiving a request for said static portion, and
 - ii. sending said static portion.
2. A method for transferring at least a portion of a resource having at least a static portion and dynamic portion over a network, comprising the steps of:
 - a. requesting at least a dynamic portion of the resource; and
 - b. receiving the dynamic portion, where the dynamic portion is adapted to be combined with the static portion.
3. The method of claim 2 further comprising the steps of:
 - a. determining whether the static portion of the resource is needed; and
 - b. if needed, requesting the static portion.
4. The method of one or more of claims 1-3 or 14, wherein said resource is a HTML document; or wherein an HTML extension relating to said resource contains at least one dynamic-variable tag; or wherein an HTML extension relating to said resource contains a loop tag; or wherein an HTML extension relating to said resource contains a tag used to define the dynamic part of the document.
5. An apparatus for transferring over a network at least a portion of a resource, the resource being made up of at least a static portion and dynamic portion, comprising:
 - a. a processor;
 - b. a memory storing instructions adapted to be executed by said processor to:
 - i. receive a request for at least the dynamic portion of the resource, and
 - ii. send the dynamic portion of the resource separate from the static portion of the resource; and
 - c. a port adapted to be connected to the network, said port and said memory coupled to said processor.
6. The apparatus in claim 5, wherein said memory storing instructions adapted to be executed by said processor further comprises:
 - a. a memory storing instructions adapted to be executed by said processor to:
 - i. receive a request for the static portion of the resource.
7. An apparatus for transferring over a network at least a portion of a resource, the resource being made up of at least a static portion and dynamic portion, comprising:
 - a. a processor;

b. a memory storing instructions adapted to be executed by said processor to:

- i. send a request for the resource;
- ii. receive the dynamic portion;
- iii. combine the dynamic portion with the static portion.

8. The apparatus of claim 7, wherein the apparatus further comprises a memory storing instructions adapted to be executed by said processor to:

- a. determine if the static portion is needed; and if needed,
- b. request the static portion.

9. The apparatus of claims 5 or 6, wherein the apparatus further comprises a memory storing said at least a portion of a resource.

10. The apparatus of one or more of claims 5-8, wherein said resource is a HTML document; or
 wherein an HTML extension relating to said resource contains at least one dynamic-variable tag; or
 wherein an HTML extension relating to said resource contains a loop tag; or
 wherein an HTML extension relating to said resource contains a tag used to define the dynamic portion of the document.

11. A computer readable medium that stores instructions adapted to be executed by a processor to perform the steps of:

- a. receiving a request from a client for at least a dynamic portion of a resource, the resource including at least a static portion and dynamic portion;
- b. sending the dynamic portion of said resource, the dynamic portion adapted to be combined with the static portion of the resource; and
- c. if the client lacks the static portion, then performing the sub-steps of:
 - i. receiving a request for the static portion, and
 - ii. sending the static portion.

12. A computer readable medium that stores instructions adapted to be executed by a processor to perform the steps of:

- a. requesting at least a dynamic portion of a resource, the resource including at least a static portion and dynamic portion; and
- b. receiving the dynamic portion, where the dynamic portion is adapted to be combined with the static portion.

13. The computer readable medium of claim 12, wherein stored instructions are further adapted to:

- a. determine if the static portion is needed; and
- b. if needed, requesting the static portion.

14. A method for transferring over a network at least a portion of a resource having at least a static and dynamic portion, comprising the steps of:

- a. receiving a request from a client for at least a dynamic portion of the resource;
- b. sending a dynamic portion of the resource in response to the request for at least a dynamic portion of the resource, the dynamic portion adapted to be combined with the static portion of said resource; and

c. after sending a dynamic portion of the resource,

i. receiving a request for the static portion of the resource; and

ii. sending the static portion of the resource in response to the request for the static portion of the resource.

5

10

15

20

25

30

35

40

45

50

55

FIG. 1A

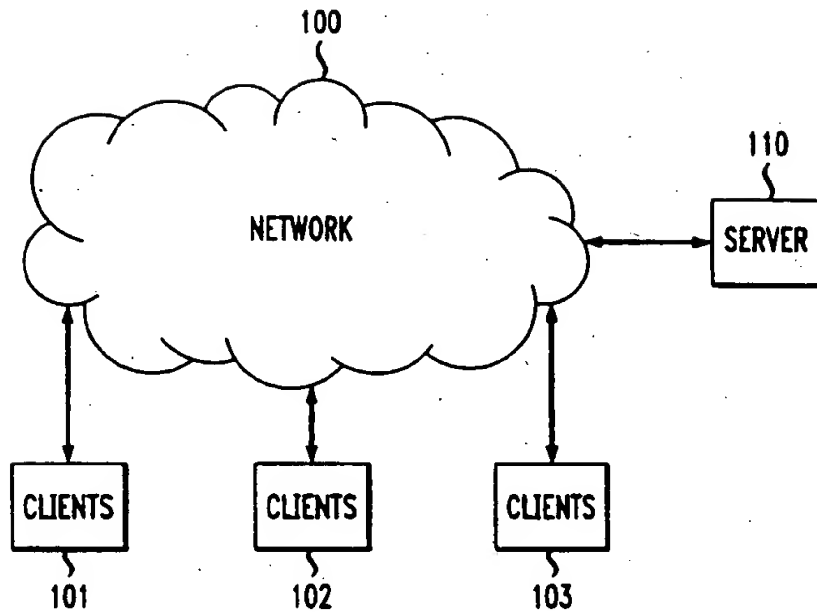


FIG. 1B

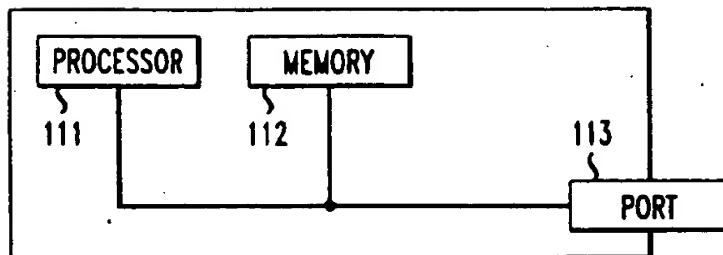


FIG. 2

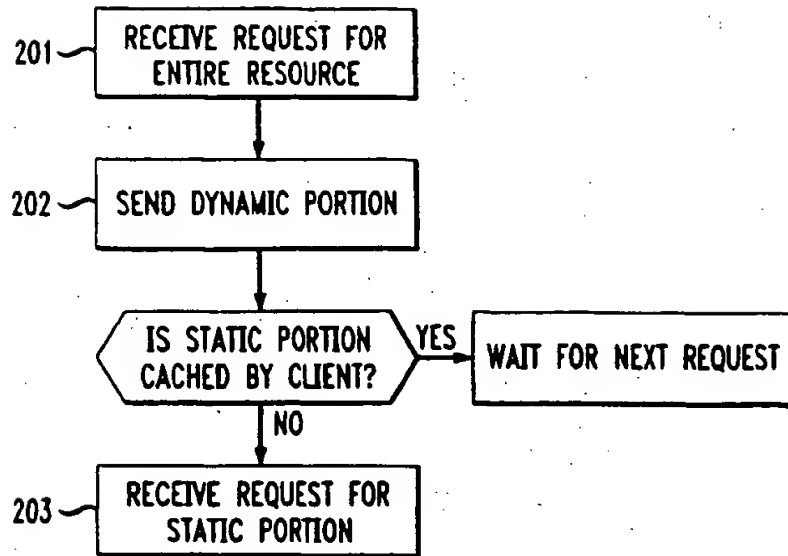


FIG. 3

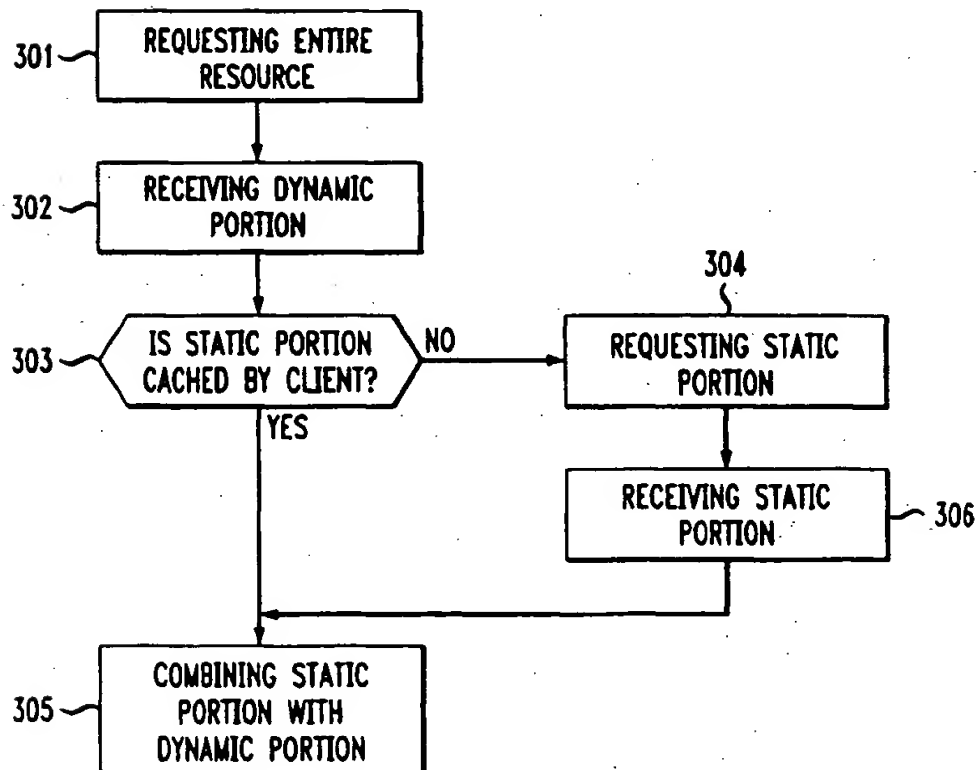


FIG. 4

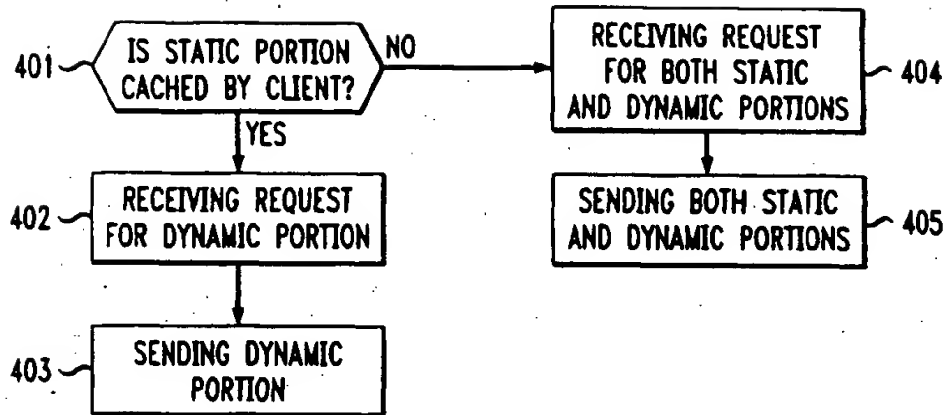


FIG. 5

